Readme

Group:
Shuk Ying Tsoi
Nat Schaffner

Explanation of file and function strategies

Search.h
- Include guard
- Function prototypes
- Data Structures

Search.c
- **insertWordNode**
  - Implementation for insertion of a word node into an index.
- **addToBack**
  - Implementation for adding a file node to a word node.
- **discoverFilename**
  - Implementation for parsing a line of filenames in a string from the inverted index into a list
- **printList**
  - Implementation for printing a list of words.
- **printList2**
  - Implementation for printing out a list of files.
- **main**
  - Implementation for:
    - checking argument number
    - opening file
    - import file to index
    - query user
- **sa**
  - Implementation for search_logical_and
  - ...Incomplete
  - Strategy:
    - Data structure for each token in the query
    - Each query word has a list of files that the word is in
      - (this can be tokenized more or less directly from the format of the inverted index file)
    - After parsing index structure for tokens and associated filenames,
      - a paring-down process occurs, as each filename starting with the first token's list is iterated through.
    - Paring-down process: if a filename is in the list for a token, it is checked against all other token's lists – when a match is found, the filename stays, but if the filename does not show up in all lists, it is removed from any list
    - After the paring down process, only the files containing all terms are left
- **so**
  - Implementation for search_logical_or

Strategy:

parse index for words matching tokens, list or results

iterate along tokens

if token == word all files are added to list

if a duplicate entry is to be inserted, it is ignored

(check if filename is contained in list)

outputAllFilenames

Implementation for moving filenames from one list to another

soAddFilename

Implementation for inserting filenames into a list